Introducción a wxPython

http://wxPython.org/

Cross-Platform GUI Library

Lic. Marcelo Fernández

marcelo.fidel.fernandez@gmail.com - http://blog.marcelofernandez.info

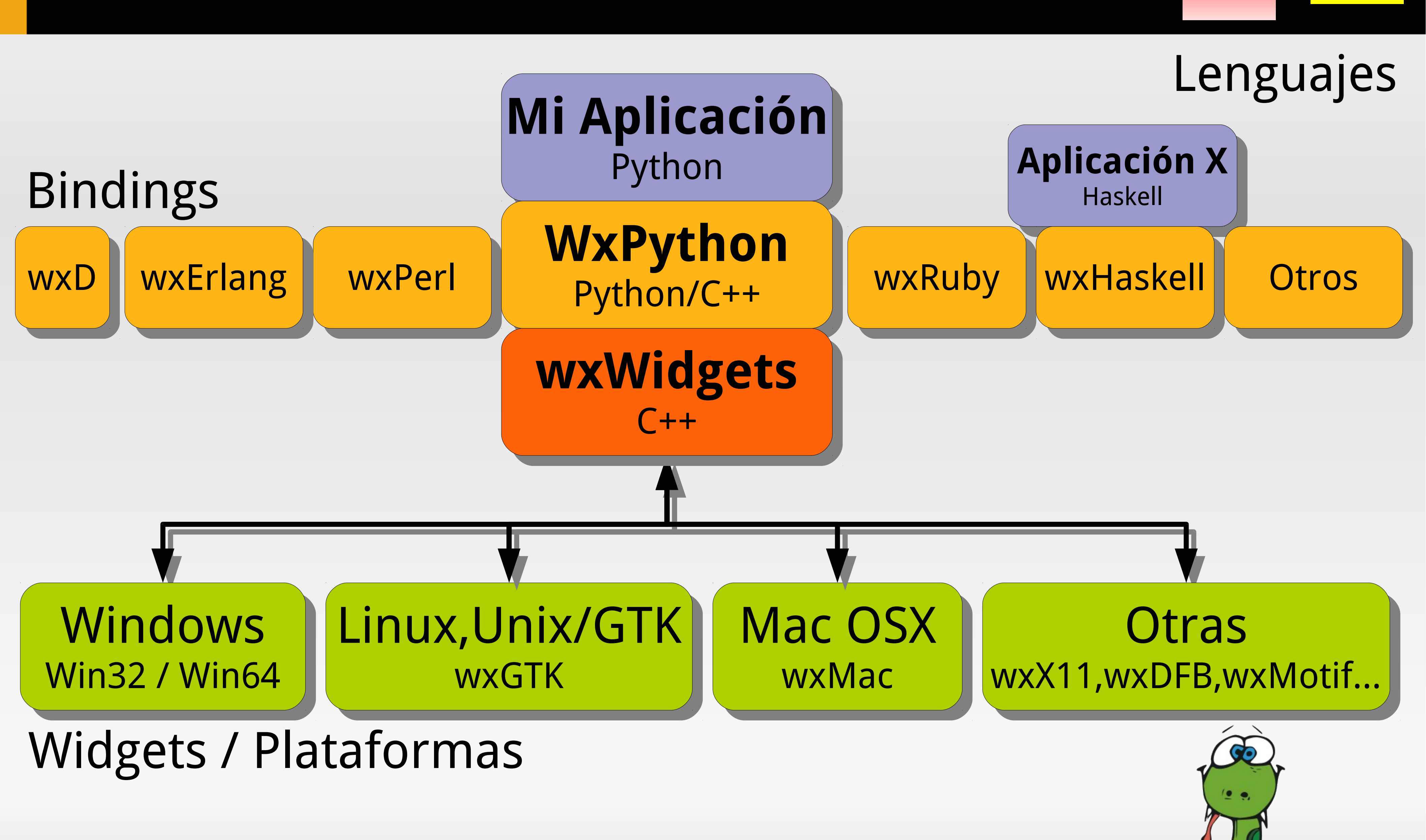
Introducción - GUIs en Python

- ¿Qué es una GUI?
- Bibliotecas: "Bindings" o "Wrappers"
- Disponibles en Python:
  - Tcl/Tk: Instalada por defecto.
  - wxPython: wrapper de wxWidgets (antes wxWindows)
  - PyGTK: wrapper de GTK.
  - PyQT: wrapper de QT.

(Ver comparativa en http://python.org.ar/pyar/InterfacesGraficas)



Introducción - wxPython



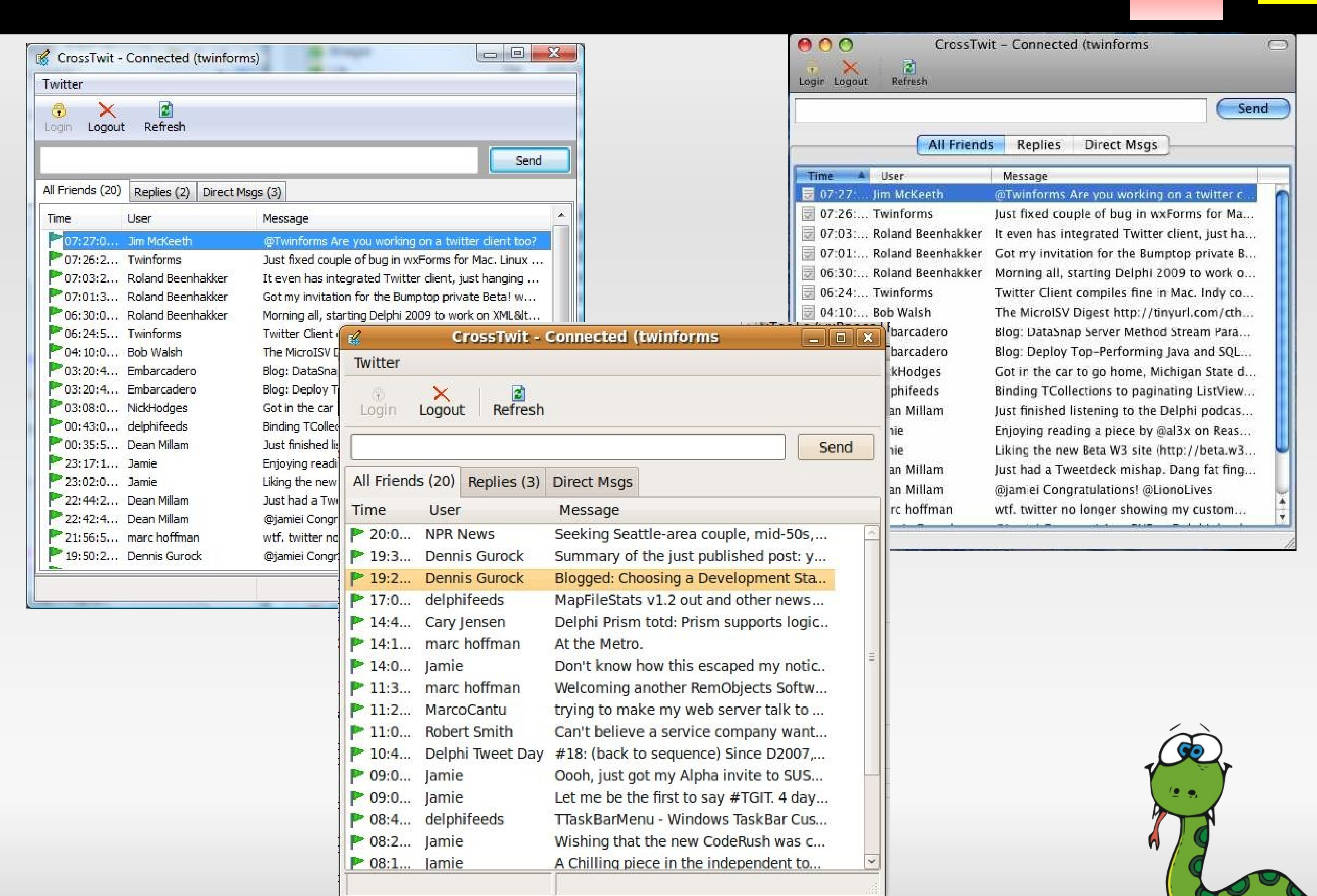
Ver más en http://wiki.wxwidgets.org/General_Information

Introducción - wxWidgets

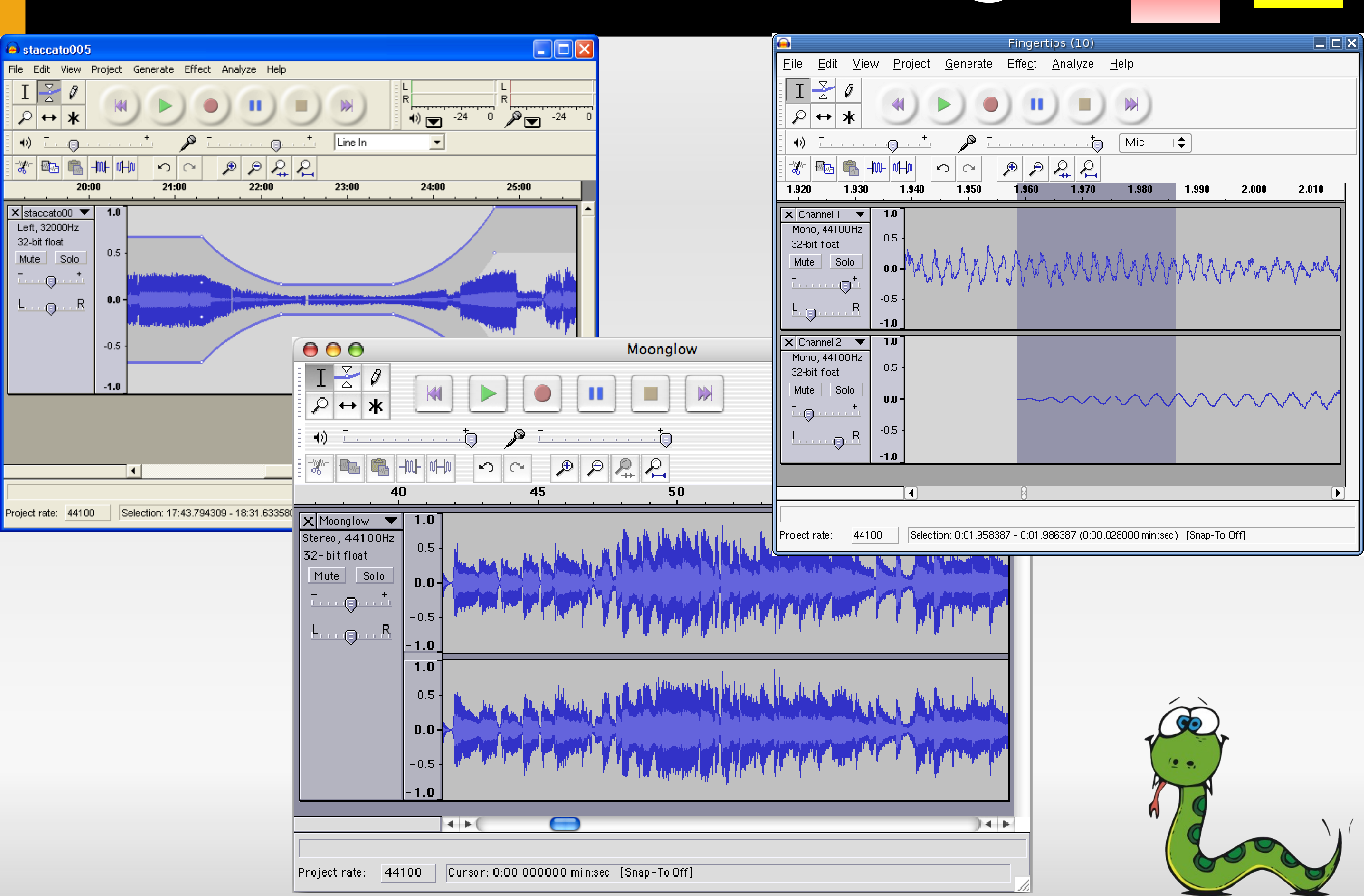
- wxPython es una biblioteca para usar **wxWidgets** (C++) desde Python
- Hereda sus características
  - Robusta, años evolucionando (1992).
  - Pensado para ser Multiplataforma desde el inicio.
  - Conserva el *Look and Feel* del entorno y su velocidad, ya que utiliza componentes GUI estándar de cada SO.
  - Permite embeber componentes nativos.
  - LGPL. Aplicaciones con cualquier licencia.
  - Windows, Linux, Mac y más con ~~casi~~ el mismo código fuente, sólo recompilando.



Introducción - wxWidgets



Introducción - wxWidgets



Introducción - wxPython

- wxPython amplía las ventajas de wxWidgets:
  - Es Python: más fácil de aprender y adecuado que C++.
  - Código ”pythónico”.
  - + Fácilmente extensible, AGW es un ejemplo.
  - Windows, Linux, Mac y más con el mismo código.
- Desventajas
  - Instalación: No está incluido en Python mismo.
  - Muchas capas de abstracción.
  - Curva de aprendizaje media/alta, lógicamente según la aplicación que se quiera desarrollar

Ejemplo 1 – Hola Mundo

```
#!/usr/bin/env python

import wx

if __name__ == '__main__':
    app = wx.App()
    frame = wx.Frame(None, wx.ID_ANY, "Hola Mundo")
    frame.Show()
    app.MainLoop()
```



Ejemplo 2 – Estructura Base

```
#!/usr/bin/env python

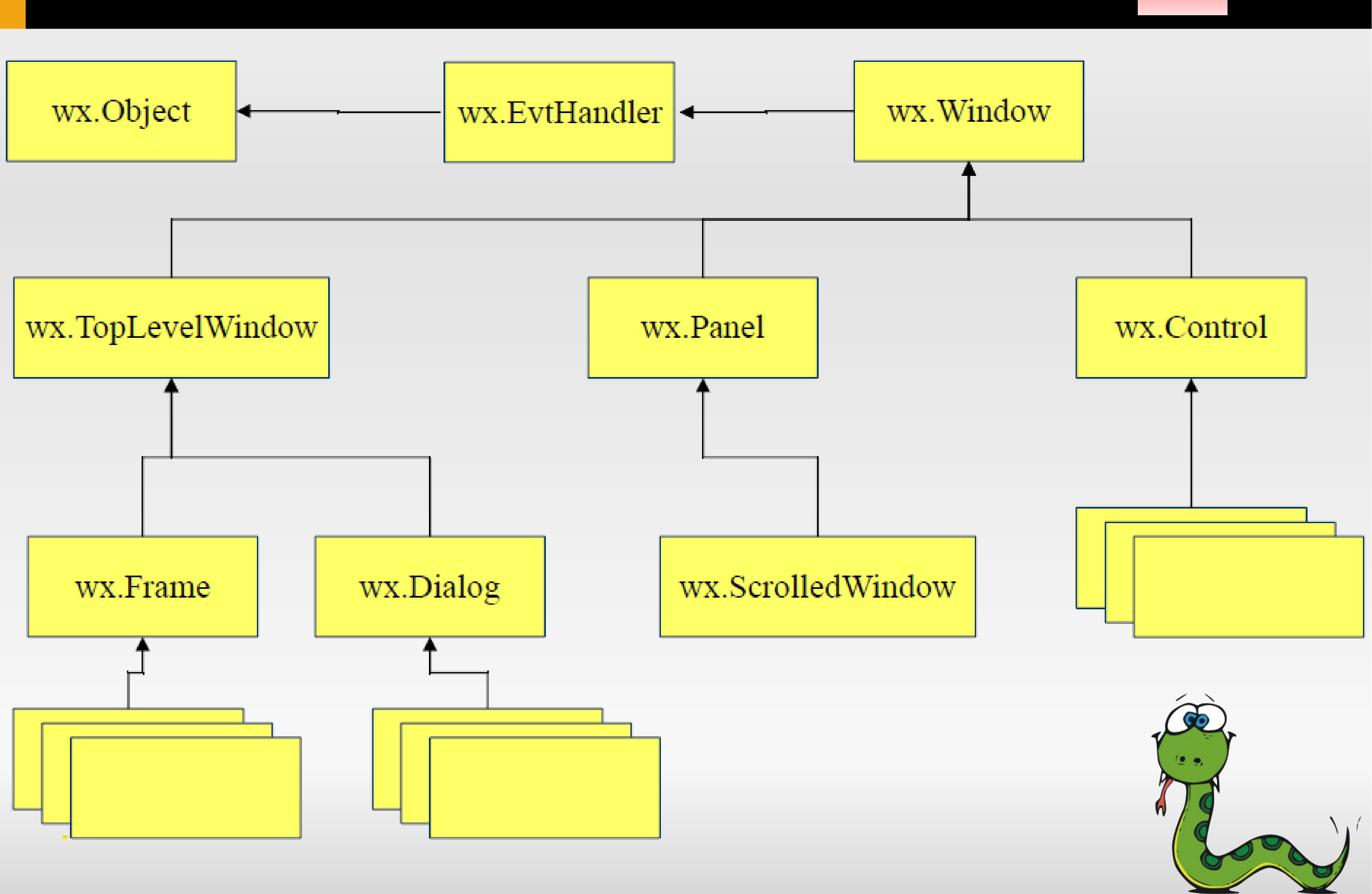
import wx

class TestFrame(wx.Frame):
    def __init__(self, parent, title):
        wx.Frame.__init__(self, parent, title=title, size=(200,100))
        self.control = wx.TextCtrl(self, style=wx.TE_MULTILINE)

if __name__ == '__main__':
    app = wx.App()
    frame = TestFrame(None, 'Editor de Texto')
    frame.Center()
    frame.Show()
    app.MainLoop()
```



Jerarquía de Clases - wxPython

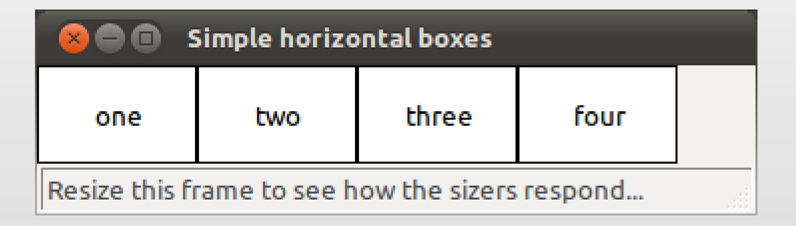


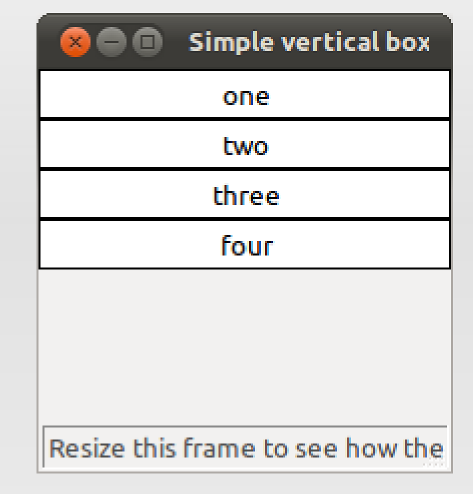
Organización de Widgets

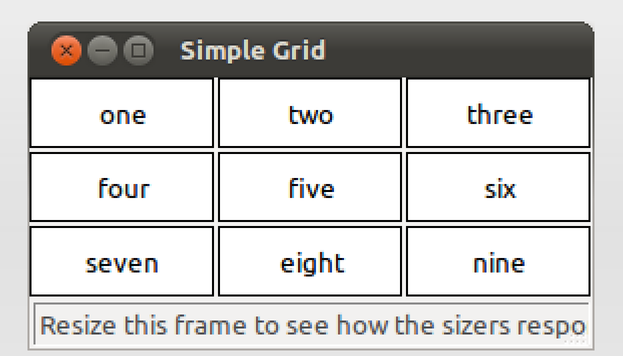
- Organización Estática
  - Posicionamiento basado en pixels
  - Gestión manual de la ubicación de los componentes
  - Pero limitado por donde se lo mire:
    - Monitores y/o Resoluciones diferentes
    - Idiomas, SOs, "Skins", Tipografías diferentes
- Organización Dinámica
  - Mucho más util en ~~todos~~ el resto de los casos
  - wxWidgets lo provee mediante los Sizers
  - Diferentes algoritmos de posicionamiento, diferentes subclases de Sizer.



Posición Dinámica - Sizers

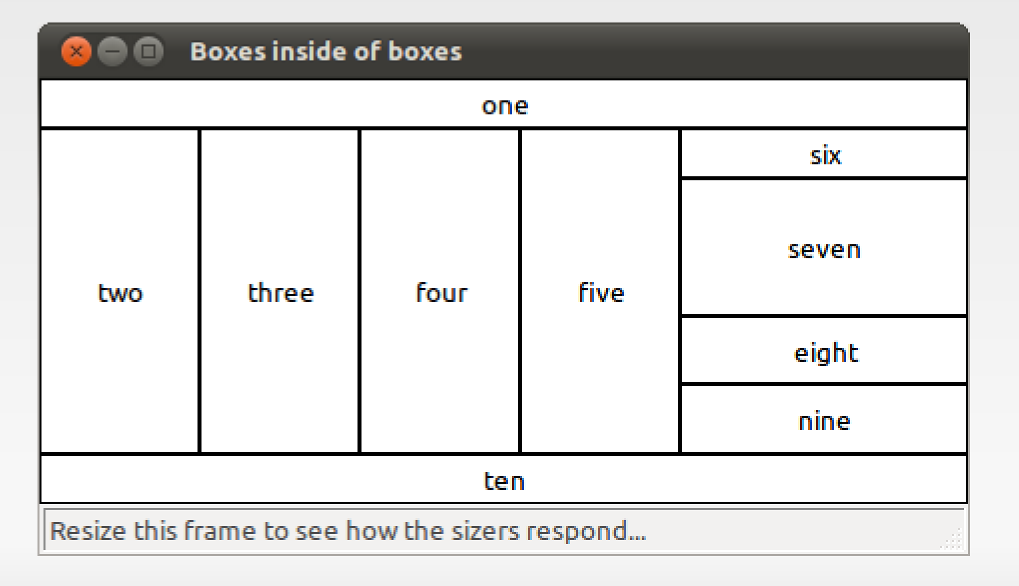


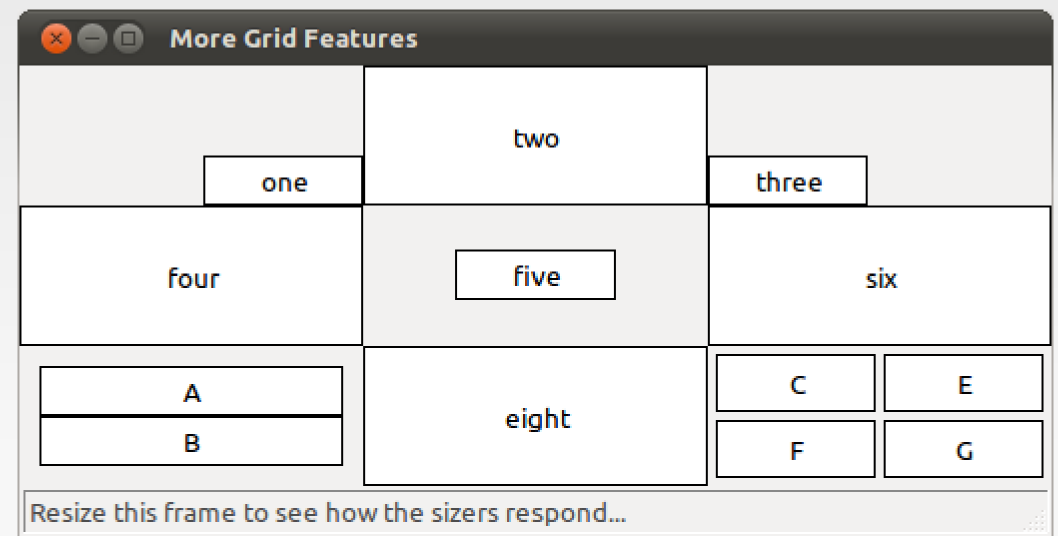




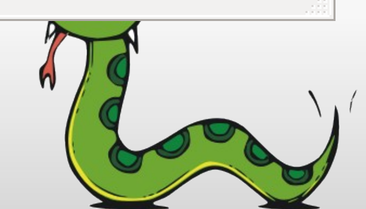
Horizontal, Vertical Boxes

Grid





Sizers combinados y anidados



Ejemplo 3 – Sizers

```
# ... Dentro de la clase MainWindow
# Sizer de Botones
self.sizer_botones = wx.BoxSizer(wx.HORIZONTAL)
# ...

# Widget -- Caja de Texto
self.txtNotes = wx.TextCtrl(self, style=wx.TE_MULTILINE)

# Sizer de Grilla
self.sizer_form = wx.GridSizer(rows=2, cols=3)

# ...

# Agrego cada Sizer al Sizer de la Ventana con su proporción y flags
self.sizer = wx.BoxSizer(wx.VERTICAL)
self.sizer.Add(self.sizer_botones, proportion=0, flag=wx.EXPAND)
self.sizer.Add(self.txtNotes, proportion=1, flag=wx.EXPAND)
self.sizer.Add(self.sizer_form, proportion=0)
self.SetSizer(self.sizer)

# ...
```

WxPython - Eventos

- wxPython está en un loop infinito, el MainLoop(), esperando que el usuario haga algo.
- Acción(Objeto) => Reacción = Evento
- Bind() permite que un evento invoque una función
- window.Bind(wx.EVT_BUTTON, self.OnClick)
- def OnClick(self, event):
    print 'Click!' # event es una instancia de wx.Event
- Siguen una jerarquía en algunos casos
- event.Skip() permite que el evento la siga



Ejemplo 4/1 - Eventos

```
  # ... Dentro de la clase MainWindow – Ejemplo Mouse

  # Defino botones Aceptar/Cancelar
  self.btnAceptar = wx.Button(self,wx.ID_ANY,u'&Aceptar')

  # Conecto el evento click a la función OnAceptar
  self.btnAceptar.Bind(wx.EVT_BUTTON, self.OnAceptar)

  # ...

def OnAceptar(self, event):
  dlg = wx.MessageDialog(self,u'¡Aceptar!',u'Test',wx.OK)
  dlg.ShowModal()
  dlg.Destroy()
```



Ejemplo 4/2 - Eventos

```
  # ... Dentro de la clase MainWindow – Ejemplo Teclado

  # Creo el widget TextCtrl
  self.txtNombre = wx.TextCtrl(self)

  # Conecto el evento Key_Up a OnTxtNombre
  self.txtNombre.Bind(wx.EVT_KEY_UP, self.OnTxtNombre)
  # ...


def OnTxtNombre(self, event):
  keycode = event.GetKeyCode()
  print u'Tecleó en Nombre: ' + str(keycode)
  if keycode == wx.WXK_TAB:
      print u'Tab!'
```

Repaso hasta aquí

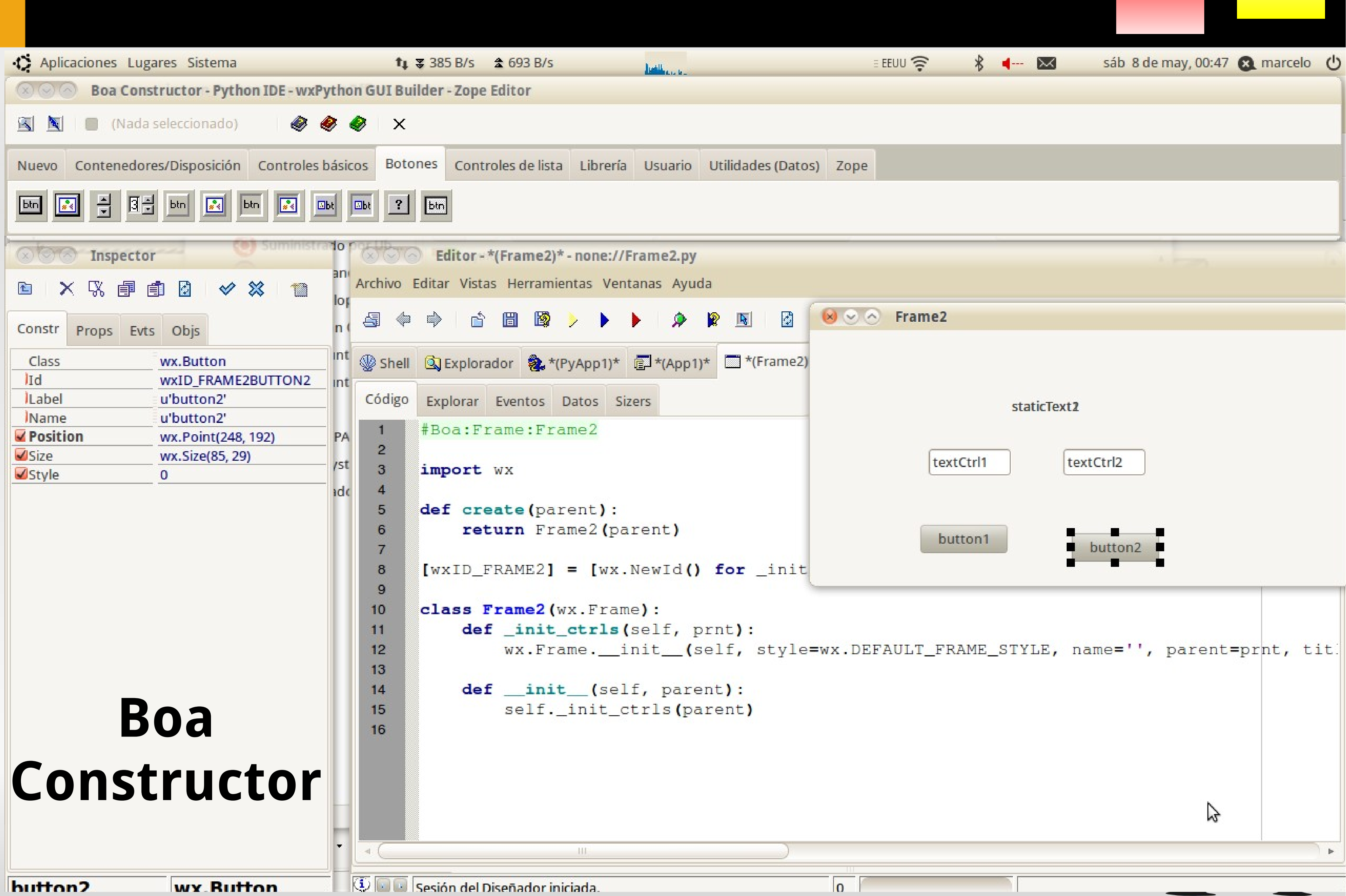
- Con poco código se pueden hacer aplicaciones de escritorio multiplataforma, sin salir de Python...
- La instalación para un desarrollador es muy sencilla y para el cliente también.
- ¡Aplicaciones con *Look and Feel* nativo!
- Lo malo: **Este método no escala**.
- La "Capa Visual" y la "Lógica de Negocio" están mezcladas. Algo no va bien...



Diseñadores de GUI

- Ventajas
  - Flexibilidad. Permiten separar el código de la vista.
  - Permiten ver inmediatamente las ventanas con nuestros widgets y sin tener que escribir código.
  - **¡Sí escala!**
  - XRC: Formato estándar de wxWidgets para describir GUIs. Es un simple archivo XML.
- Desventajas
  - No sirve en todos los casos: Formularios dinámicos.
  - La carga del XML es un poco más lenta que si armamos la interfaz con código.

Boa Constructor



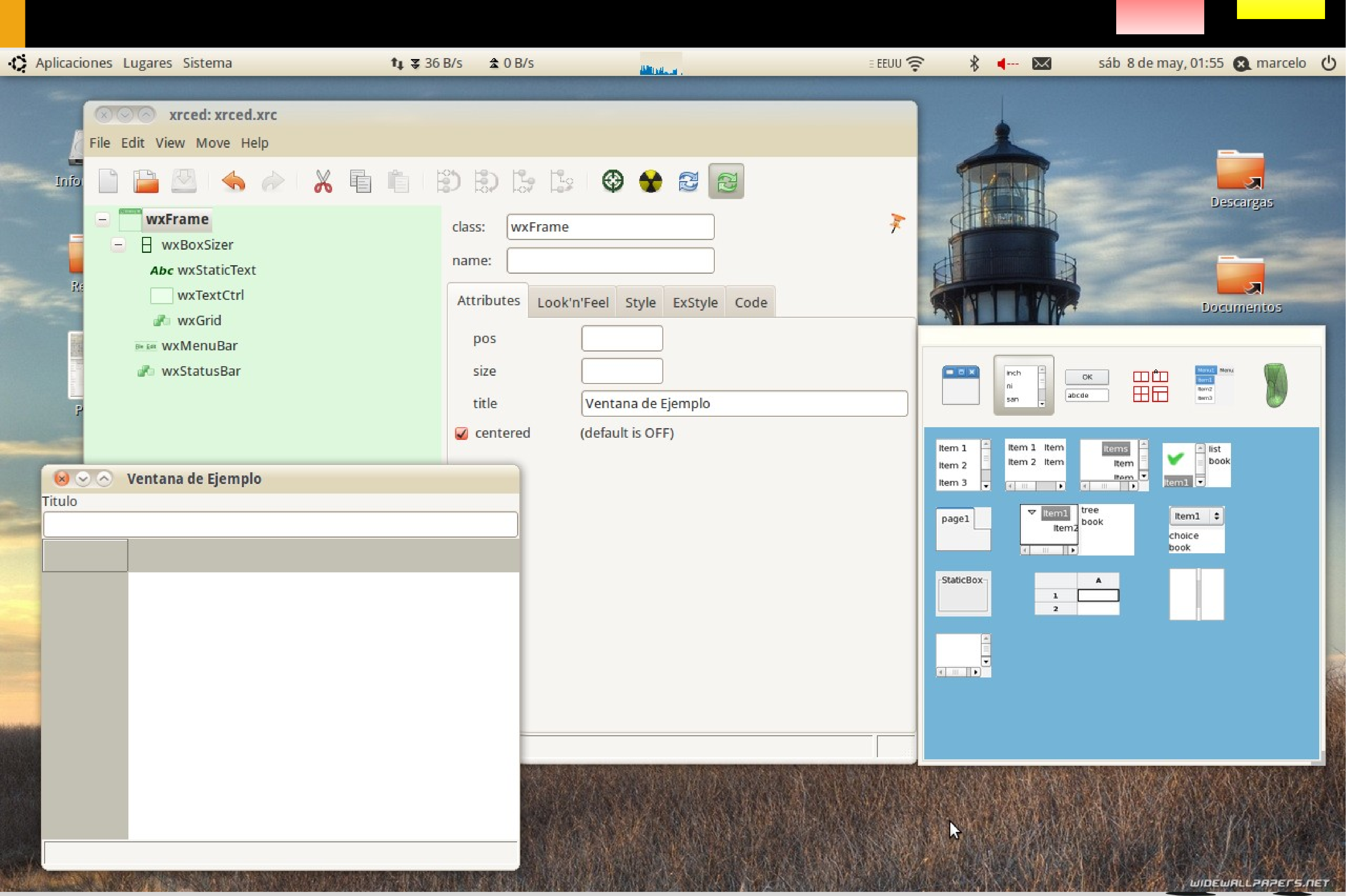
wxGlade

wxGlade

- Totalmente Visual
- Genera XRC y .py
- No soporta muchos de los controles actuales de wxPython
- Bastante rústico

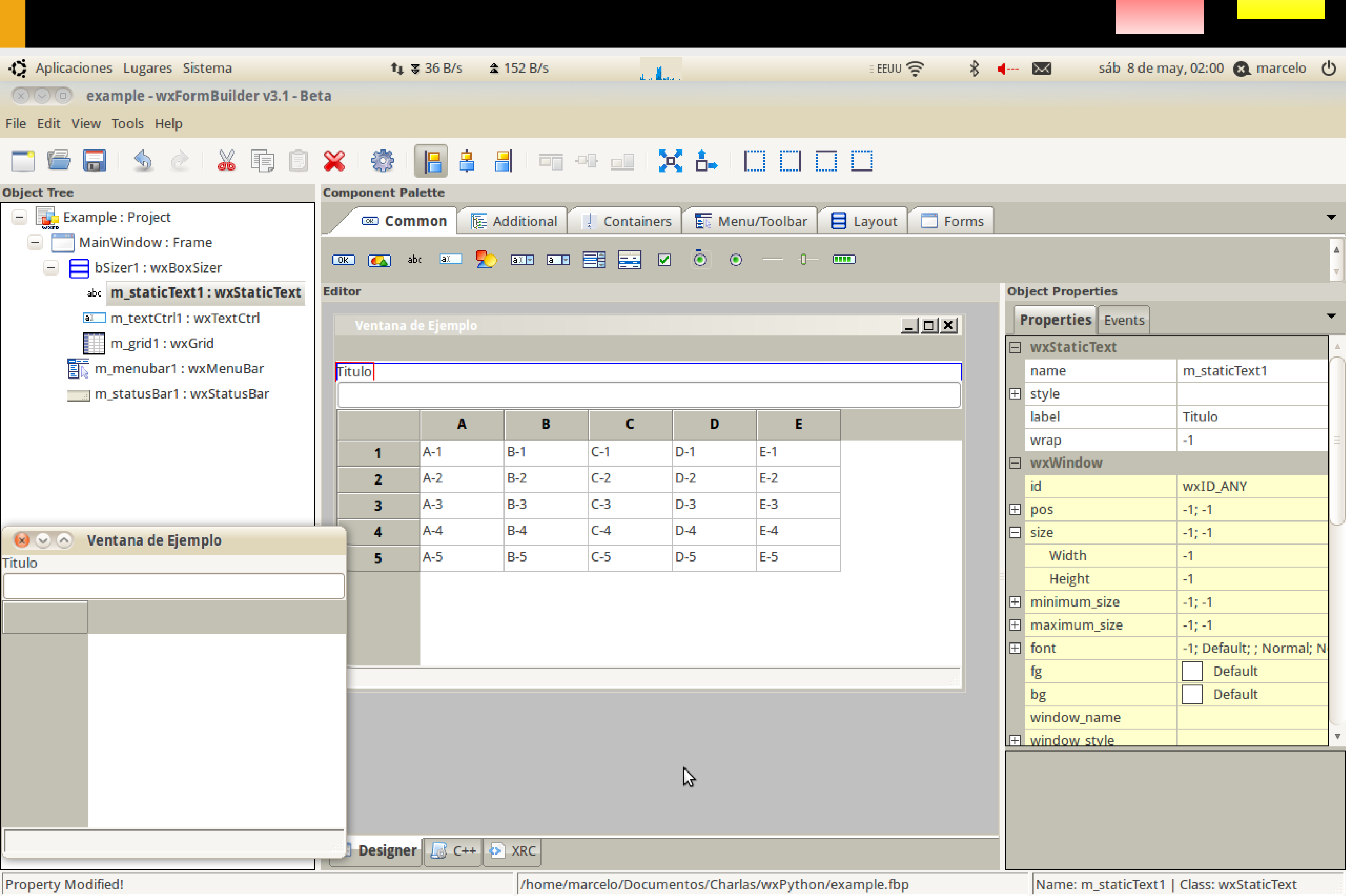


XRCed



Aplicaciones Lugares Sistema
36 B/s
0 B/s
EEUU
sáb 8 de may, 01:55
marcelo
xrced: xrced.xrc
File Edit View Move Help
wxFrame
wxBoxSizer
Abc wxStaticText
wxTextCtrl
wxGrid
wxMenuBar
wxStatusBar
class: wxFrame
name:
Attributes
Look'n'Feel
Style
ExStyle
Code
pos
size
title
Ventana de Ejemplo
centered
(default is OFF)
Ventana de Ejemplo
Titulo
Descargas
Documentos
Item 1
Item 2
Item 3
page1
tree
book
Item1
Item2
choice
book
list
book
StaticBox
WIDEWALLPAPERS.NET

wxFormBuilder



Ejemplo 5 – XRC

¡Con esta línea armé todo el form!

```
import  wx
import  wx.xrc as xrc

class EditorApp(wx.App):

    def OnInit(self):
      # Cargo el XRC y el frame principal
      self.res = xrc.XmlResource('example.xrc')
      self.frame = self.res.LoadFrame(None, 'MainWindow')

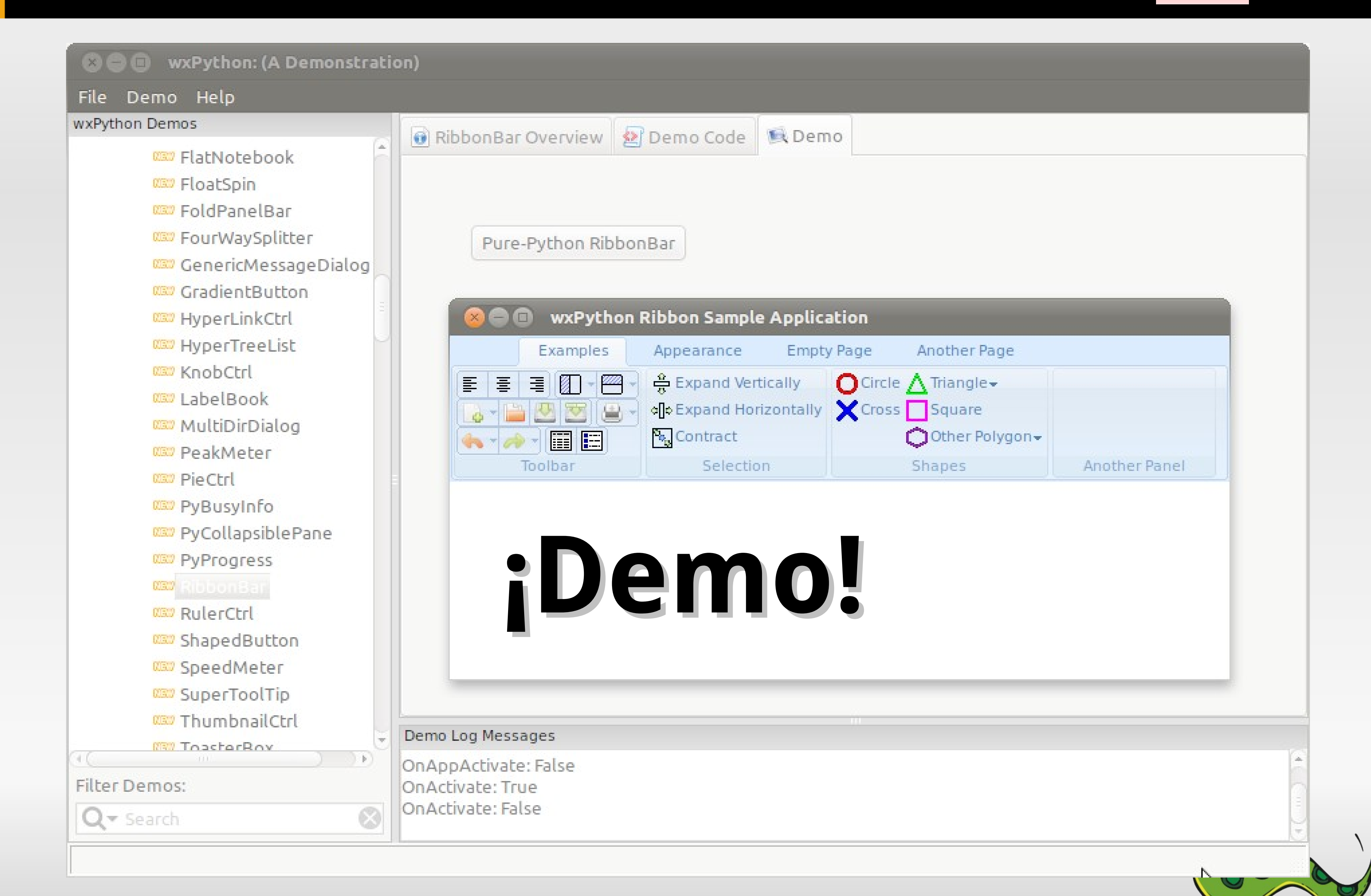
      # Obtengo la referencia al texto, útil
      self.txtTexto = xrc.XRCCTRL(self.frame, 'txtTexto')

      # Conecto los eventos ...
      self.frame.Bind(wx.EVT_MENU,self.OnSalir,id=xrc.XRCID('mnuSalir'))
      self.frame.Show()
      return True

    # ... Defino las demás funciones / callbacks en la App

if __name__ == '__main__':
    app = EditorApp()
    app.MainLoop()
```

wxPython Demo



Y como si fuera poco...

- Visualización e impresión de HTML simple
- Print Framework, con vista previa y configuración
- Clipboard y drag and drop
- Ayuda en línea. Gran comunidad alrededor
- Librería de graficación de objetos vectoriales: OGL.
- Soporte para Cairo y OpenGL (GLCanvas)
- Texto enriquecido (RTF) y "estilizado" (STC)
- Animaciones y multimedia
- Programación multiproceso, Unicode, componentes personalizados, wx.AUI



Links

- Sitio oficial: http://www.wxpython.org
- Libro de referencia: wxPython in Action (Manning)
- Wiki Comunidad: http://wiki.wxpython.org
- Listas de correo:
  - wxPython-users
  - wx-users
  - PyAr - http://www.python.com.ar ;-)
- http://www.zetcode.com/wxpython/
- http://pablotilli.com.ar/

